AFRL-IF-RS-TR-2004-263
Final Technical Report
September 2004

SEMANTIC INTEROPERABILITY MEASURE: TEMPLATE-BASED ASSURANCE OF SEMANTIC INTEROPERABILITY IN SOFTWARE COMPOSITION (TBASSCO)

University of Southern California at Marina del Rey

Sponsored by
Defense Advanced Research Projects Agency
DARPA Order No. K513

APPROVED FOR PUBLIC RELEASE; DISTRIBUTION UNLIMITED.

AIR FORCE RESEARCH LABORATORY
INFORMATION DIRECTORATE
ROME RESEARCH SITE
ROME, NEW YORK

STINFO FINAL REPORT

This report has been reviewed by the Air Force Research Laboratory, Information Directorate, Public Affairs Office (IFOIPA) and is releasable to the National Technical Information Service (NTIS). At NTIS it will be releasable to the general public, including foreign nations.

AFRL-IF-RS-TR-2004-263 has been reviewed and is approved for publication

APPROVED: /s/

RAYMOND A. LIUZZI
Project Engineer

FOR THE DIRECTOR: /s/

JAMES A. COLLINS, Acting Chief
Information Technology Division
Information Directorate

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 074-0188

1. AGENCY USE ONLY (Leave blank)

2. REPORT DATE
SEPTEMBER 2004

3. REPORT TYPE AND DATES COVERED
Final Jun 00 – Sep 03

4. TITLE AND SUBTITLE
SEMANTIC INTEROPERABILITY MEASURE: TEMPLATE-BASED ASSURANCE OF SEMANTIC INTEROPERABILITY IN SOFTWARE COMPOSITION (TBASSCO)

5. FUNDING NUMBERS
C - F30602-00-2-0610
PE - 62301E
PR - DASA
TA - 00
WU - 12

6. AUTHOR(S)
Ke-Thia Yao, Robert Neches,
In-Young Ko, and Robert MacGregor

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
University of Southern California at Marina del Rey
Information Sciences Institute
4676 Admiralty Way
Marina del Rey California 90292-6695

8. PERFORMING ORGANIZATION REPORT NUMBER
N/A

9. SPONSORING / MONITORING AGENCY NAME(S) AND ADDRESS(ES)
Defense Advanced Research Projects Agency
3701 North Fairfax Drive
Arlington Virginia 22203-1714

AFRL/IFTB
525 Brooks Road
Rome New York 13441-4505

10. SPONSORING / MONITORING AGENCY REPORT NUMBER
AFRL-IF-RS-TR-2004-263

11. SUPPLEMENTARY NOTES

AFRL Project Engineer: Raymond A. Liuzzi/IFTB/(315) 330-3577/ Raymond.Liuzzi@rl.af.mil

12a. DISTRIBUTION / AVAILABILITY STATEMENT
APPROVED FOR PUBLIC RELEASE; DISTRIBUTION UNLIMITED.

12b. DISTRIBUTION CODE

13. ABSTRACT *(Maximum 200 Words)*
SIM-TBASSCO (semantic Interoperability Measures: Template-Based Assurance of Semantic Interoperability in Software Composition) addresses a longstanding software engineering goal of assembling software from components. Conventional approaches support composition up to a point, but cannot handle qualitative considerations in composition, such as implementation effort, performance, resource requirements, or reliability. SIM-TBASSCO helps software developers engage in guided, efficient searches and evaluations of the set of alternative system implementation that can be built with the components available to them. It will let developers evaluate components' functional and data equivalence compatibility and find pertinent data conversion mappings. During runtime SIM-TBASSCO helps system administrators monitor and gauge the health of system and enables them to perform dynamic reconfigurations to overcome bottlenecks/faults while preserving design time constraints.

14. SUBJECT TERMS
Knowledge Base, Databases, Artificial Intelligence, Software, Information Systems

15. NUMBER OF PAGES
20

16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT	18. SECURITY CLASSIFICATION OF THIS PAGE	19. SECURITY CLASSIFICATION OF ABSTRACT	20. LIMITATION OF ABSTRACT
UNCLASSIFIED	UNCLASSIFIED	UNCLASSIFIED	UL

Table of Contents

1 Summary 1
2 Introduction 1
3 Methods, Assumptions, and Procedures 1
3.1 Testbed application 1
3.1.1 GeoWorlds 1
3.1.2 GeoTopics 3
3.2 Metadata level modeling 4
3.2.1 Content and structural forms 5
3.3 Semantically-based service scripting 5
3.3.1 Model for representing active document collection scripts 6
3.3.2 Script instantiation and execution 6
3.3.3 Service scripting tool 7
3.4 Software Gauges 8
3.4.1 Interoperability gauge 8
3.4.2 Compatibility gauge 9
3.4.3 Insertion gauge 10
4 Results and Discussion 12
References 15

List of Figures.

Figure 1 GeoTopics web page displaying the daily top 20 hot topics and hot places (with map)……………………………………………………………………………..3
Figure 2 Schema model for representing document collection and service semantics…...4
Figure 3 Schema model for representing active document collection scripts…………..…6
Figure 4 Service Scripting Tool………………………………………………………..6
Figure 5 Compatibility Gauges…………………………………………………………...9
Figure 6 Insertion Gauges………………..……………………………………………...11

1 Summary

SIM-TBASSCO (Semantic Interoperability Measures: Template-Based Assurance of Semantic interoperability in Software COmposition) addresses a longstanding software engineering goal of assembling software from components. Conventional approaches support composition up to a point, but cannot handle qualitative considerations in composition, such as implementation effort, performance, resource requirements, or reliability. SIM-TBASSCO helps software developers engage in guided, efficient searches and evaluations of the set of alternative system implementations that can be built with the components available to them. It will let developers evaluate components' functional and data equivalence compatibility and find pertinent data conversion mappings. During runtime SIM-TBASSCO helps system administrators monitor and gauge the health of system and enables them to perform dynamic reconfigurations to overcome bottlenecks/faults while preserving design time constraints.

2 Introduction

The SIM-TBASSCO project is developing a metadata framework for describing software components that supports the dynamic assembly and reconfiguration of software systems. During design-time the framework supports *semantic-level gauges* that help application developers to select and combine interoperable software components. This facilitates rapid composition of semantically validated software architectures as components are assembled into special-purpose applications. A *semantically-based scripting tool* helps users design a data-flow style architecture, and helps users to incrementally modify, instantiate and test the architecture by allocating correct resources. During runtime, the framework supports *software architecture views* that provide visibility into software systems and that identify control points to adjust their behavior to dynamically and rapidly respond to faults. Multi-level views offer a greater range of adjustments than any single level. A *system architecture view* enables dynamic adjustment of servers: create additional server to accommodate increased demand, and migrate server from overloaded host to new host. A *dataflow architecture view* enables reformulation of an application: substitute alternative type of service for non-functioning or unavailable service. This ability to self-monitor and to reconfigure on-the-fly enables self-healing and affords increased performance.

3 Methods, Assumptions, and Procedures

3.1 Testbed application

3.1.1 GeoWorlds

As a testbed application for this work, we have adopted GeoWorlds, a component-based Web and geographic information management system. GeoWorlds is in use at US Pacific Command (USPACOM), where it is used by analysts at the Virtual Information Center, as well as by USPACOM's Crisis Operations Planning Team. The system is also in use at

Joint Forces Command (JFCOM), where it is currently under consideration by the Joint Battle Center for deployment throughout JFCOM as an interim capability. End users and application developers create special-purpose analysis services by drawing on components in the GeoWorlds system.

At USPACOM GeoWorlds operates both in the unclassified domain as well as on the classified SIPRNet in the PACOM Crisis Operations Center SCIF. The Crisis Operations Planning Team is responsible for tracking current events throughout USPACOM's 13 million square mile region of responsibility, monitoring situations that could potentially expand into crises requiring US military action, and preparing and maintaining detailed force deployment plans in anticipation of likely requirements. Its leadership reports to the USPACOM Deputy Commander in Chief (DCinC) for Operations (J5). The Virtual Information Center is a matrixed organization underneath PACOM's Commander in Chief, serving both the DCinC for Operations and the DCinC for Intelligence (J2). Its responsibility is to complement the traditional intelligence process with rapid access and analysis of open source data available over the Internet.

3.1.2 GeoTopics

As a second test-bed application we have developed GeoTopics (http://www.isi.edu/geoworlds/geotopics/), a HTML-based application build using GeoWorlds components. Previously, GeoWorlds has relied upon search engines to retrieve information. But, based upon our experience immediately after the September

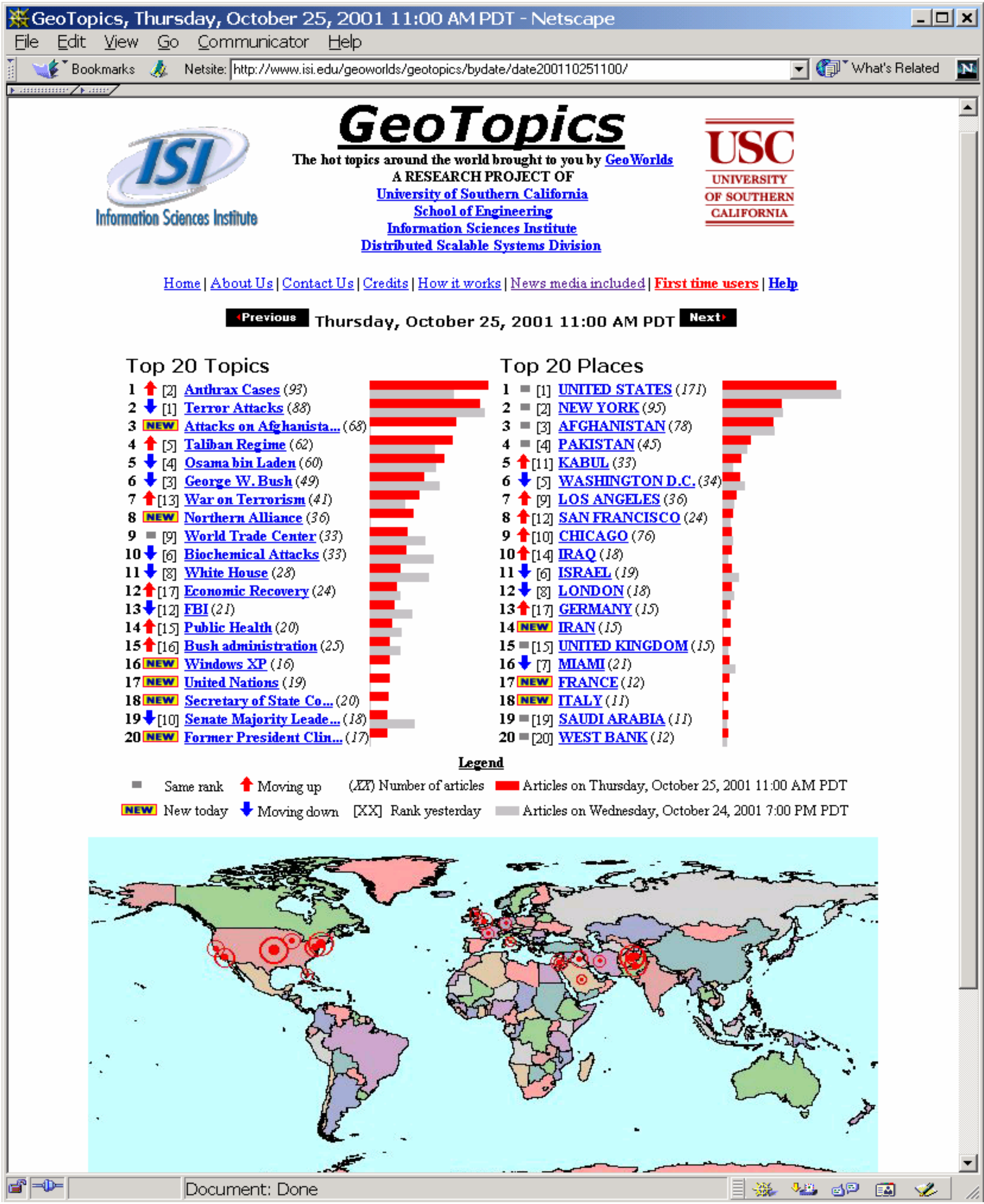


Figure 1GeoTopics web page displaying the daily top 20 hot topics and hot places (with map)

11th attack search engines are not capable of keeping up with fast changing and continuously developing information. Although most search engines attempted to index news articles throughout the day, most of the information returned on September 11th are either not relevant or obsolete. This prompted us to develop GeoTopics (see Figure 1), a web-based service that directly monitors daily news articles from a number of large, elite English-language newspaper websites. GeoTopics contributes by helping to identify the "hot topics," and the most frequently referenced places, found that day in this very large collection of reports. The goal is to help users look at what's going on worldwide in either of two ways: *what places are relevant to a topic,* or *what topics are hot in a place*.

GeoTopics provides a second testbed option to the members of the IntelliGauge TIE. The members still can use the Burma Drug scenario, if they prefer that application. GeoTopics presents a variety of interesting challenges, as well as points of integration and collaboration:

- Unlike GeoWorlds, which in some respects may be viewed as a component framework, GeoTopics is a component application. This application executes daily to gather news articles over the web and analyzes them. This provides an opportunity for the runtime members of the IntelliGauge TIE to probe and monitor the performance of the application.
- GeoTopics gather news articles from newspaper sites over the web, so it is subject to problems created by network congestions/interruptions and by overloaded websites (during heavy news days). Currently, GeoTopics simply throws Java exceptions when these problems occur. Object Services' Prospector should be an ideal fit in monitoring this kind of Java VM level problems.
- Some of GeoTopics' services are executed remotely on distributed servers. Currently, problems with overloaded servers or unresponsive servers are ignored. Columbia CHIME's distributed probe architecture should be an ideal fit in monitoring this kind of remote server problems.
- With respect to SIM-TBASSCO, GeoTopics offers a new set of problems to stress the scripting tool , and it offers additional opportunities for developing new architectural gauges .

3.2 Metadata level modeling

A vision of component-based software development is to provide a repository of software components for application developers to select from to rapidly build their product. In order for this vision to work the application developers must have confidence that the selected components are meaningfully interoperable. Component interface specifications, such as IDL, do not capture enough information to ensure that the semantic intents are satisfied during software composition. Here we propose a metadata level model to capture and reason about this semantics.

The insufficiency of interface-based specification is especially noticeable in the case of GeoWorlds, where a uniform service component API is adopted, and a common document collection data structure is used for most of GeoWorlds' information management service components. Within GeoWorlds many services can be forced syntactically together with no compilation error. Although this provides a very flexible mechanism to combine services, when used incorrectly it may generate run-time exceptions or strange results.

3.2.1 Content and structural forms

We have adopted a lightweight, multi-form ontology to present the semantics of document collections. This model is designed to capture essential aspects of components that are of interest to the developer, while omitting semantic forms that would prevent the system from performing quick inference. Based on our experience with GeoWorlds, this lightweight representation is sufficient for most *current* Web-based information management services.

The semantic specification of a document collection can be divided into two forms: *content* and *structure*. The *content description* represents the contextual meaning of the collection (e.g., a document collection in which the documents are classified by the major noun phrases). The *structure description* characterizes the organization structure (e.g., a document collection organized in an acyclic graph structure).

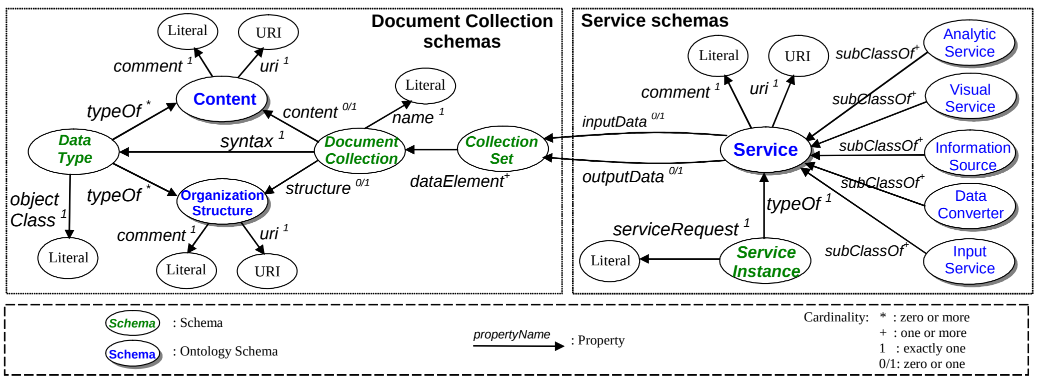


Figure 2 Schema model for representing document collection and service semantics

Domain-specific ontologies are used to discriminate and classify the document collection content types and organization structures, and service functionalities. A schema model has been developed to represent the document collection and service semantics in terms of its content, organization structure, and active relations between other document collections. Figure 2 illustrates this schema model as an ER (Entity-Relationship) diagram. Content, Organization Structure, and Service are the top-level ontology schemas that describe the top concepts in the ontology hierarchies of instantiated schemas, which we call nodes. Subsumption relations between nodes can be described by subClassOf property. In the current implementation, Analytic Service, Visual Service, Information Source, Data Converter, and Input Service are defined as the major service types that are subclasses of Service.

3.3 Semantically-based service scripting

The *Service Scripting Tool* allows application developers and end users to combine multiple services together to perform complex information analyses. This tool provides a GUI for visually composing a service script by means of a data flow diagram. During the service scripting time, this tool uses the metadata level model to ensure that users only select interoperable services to create semantically well-formed scripts. Scripts can either be concrete (specifying specific services to use) or abstract (using metadata to specify classes of services to use). Abstract scripts can be dynamically instantiated at run-time based on components available from local component repositories. This dynamic instantiation not only allows for adaptation of the script to system environment changes, but also it allows for adaptation to utilize components newly added to component repositories.

3.3.1 Model for representing active document collection scripts

The scripting tool requires semantic descriptions that model the behavior of components it operates on. The model previously described for representing semantics of document collections and services has been extended to provide a model to represent active document collection scripts and their instances. Figure 3 shows the ER diagram of this extended model.

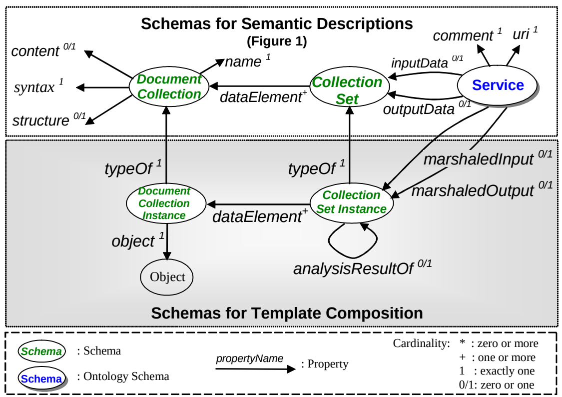


Figure 3 Schema model for representing active document collection scripts

When an information service is matched against a set of document collections, the collections within the set should be bound to the service as marshaled input parameters, so that the service can be performed by using the data at run-time. Also, when a service is combined (pipelined) with another service, the output document collections of the first service should be bound to the input parameters of the second service. *marshaledInput* and *marshaledOutput* properties of Service schema represent such data bindings between nodes. Using Collection Set Instance schema, which elements are Document Collection Instance's, can represent a set of document-collection instances. An *analysisResultOf* property in a Collection Set Instance explicitly represents the input-output relationship between the set and another document-collection set via a service. Each Document Collection Instance has an *object* property which points to the physical object that keeps the content of the document collection.

A document collection can be a member of multiple I/O data sets and it can participate within multiple document relations. For example, consider a document collection composed of Spanish documents and a document collection categorized based on place names cited in the document contents. These document collections are the results of an English-to-Spanish translation service and a place name extraction service performed on an initial document collection.

3.3.2 Script instantiation and execution

An active document collection script can be instantiated by allocating local resources to

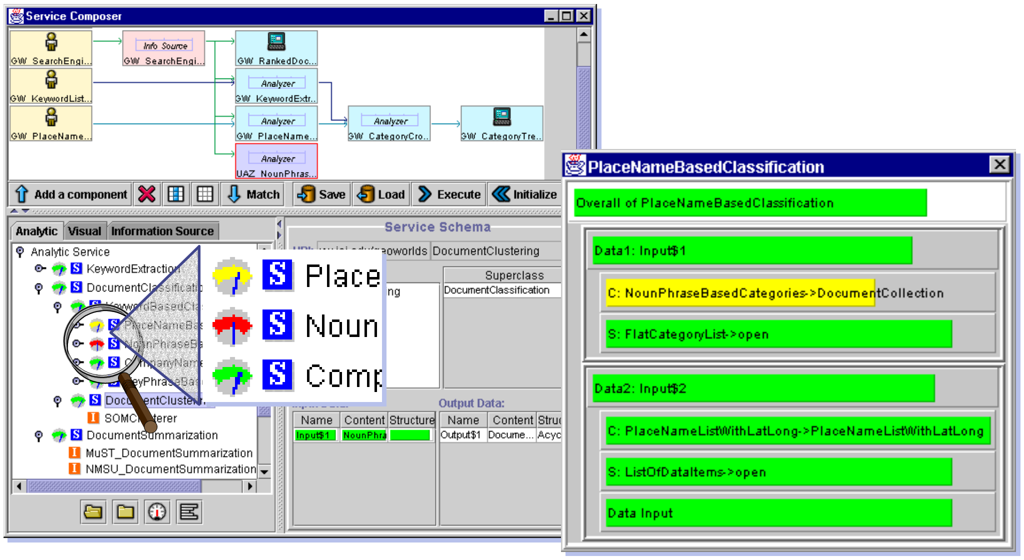


Figure 4a. Service Scripting Tool (left); b. Compound Interoperability Gauge (right)

the nodes in the script. In a local system, semantic descriptions about the local resource instances are kept as metadata in a repository. Metadata about a resource instance also includes some syntactic information such as data types, job request entries, and I/O parameter ordering. The semantic compatibility measurements (explained at Section 5) are performed to select semantically compatible local resources for each node in the script. The process of instantiating a script may require human interaction to resolve multiple matches of resource instances and syntactic mismatches between nodes.

As the result of an instantiation, proxies are created to act as clients to invoke the specific services that are selected to instantiate the script, and to receive the results of these service instances. Each *proxy* delegates a resource instance (a document collection or a service) and keeps information for accessing the local resource. For each functional semantics, a service proxy is created and for each document-collection set, an ordered-list (ordered based on the I/O parameters of the corresponding service instance) of document collection proxies is created. A service proxy keeps precedence relationship between predecessor and successor proxies and a pointer to the corresponding semantic description in the script. A document collection proxy maintains a pointer to a document collection object.

The instantiated script can be executed by running the service proxies in a sequence governed by the precedence relations. An activated service proxy submits a service request to the system interface, monitors the job status, and receives the result. In the current prototype implementation, this service access mechanism is implemented based on the GeoWorlds' asynchronous service invocation architecture.

3.3.3 Service scripting tool

The active document collection script composer provides GUI-based tools to help users set up analysis and structuring activities by composing, instantiating and executing scripts. Figure 4a) shows the script composer window. The upper part displays the current script including services and connections between them. Given a selection of nodes in the script, the lower part shows the partial ontology hierarchies that are semantically compatible with the selected nodes. This lets users see what options are available to them for adding steps to their analysis.

As described in the previous section, the composer creates proxies and proxy connections (a directed acyclic graph of control and data flow between proxies) when a script is instantiated. This enables the system to invoke, monitor and synchronize the services. Also, with the help of the inference engine, it automatically finds and inserts syntactic converters (e.g., data type converters) between syntactically mismatched components if appropriate converters are available. When a script is executed, each node in the graph displays the status (progress bar and messages) of the service.

3.4 Software Gauges

The metadata level modeling and the service scripting mechanism described in the previous sections provide the underpinnings for a functional software component repository. Here we describe a set of software gauges that facilitates the usage and maintenance of the repository. Our current work supports gauges geared towards three types of repository users:

- *Interoperability gauges* for application developers to efficiently search for candidate components in the script generation process.
- *Compatibility gauges* for system administrators to adapt scripts to their local system environments by finding candidate replacement components.
- *Insertion gauges* for component developers to judge the level of effort required to insert their component into the repository, and determine the uniqueness of their component within the repository.

Currently, the main metric the software gauges use is the semantic distance based on graph distance in the representation we use to model components. Given two data schemas in a taxonomy hierarchy, this metric determines if one schema subsumes (is the ancestor of) the other schema. If it does subsume, it determines the graph distance of the two schemas in terms of their content and structure. Finally the metric returns a value inversely proportional to the graph distance.

3.4.1 Interoperability gauge

The Service Scripting Tool allows users to script semantically correct service data flow diagrams. At each step of the service construction process, the tool offers a set of semantically interoperable services for the user to choose. However, for the inexperienced user the number of choices can be confusing and overwhelming. To help a user to make appropriate selection we have implemented a kind of Scripting Gauge called the Interoperability Gauge. This gauge uses the graph distance metric to rank the services based on their data compatibility (from output data schema to the input data schema of the connecting service). This gauge favors more specialized services over more generic services. For example, the output clusters of the Self-Organizing MAP (SOM) clustering service can be displayed by a generic hierarchical category visualization tool. However, there is a 2D SOM map visualization tool designed especially to visualize the clusters. The Interoperability Gauge ranks the map visualization tool higher than the category visualization tool.

We have developed several ways to visualize the interoperability gauge results, as shown in Figure 4a and 4b. We have augmented our service selection panel (bottom left portion of the Service Scripting Tool panel in Figure 4a by displaying a dial gauge next to each service choice summarizing its interoperability. The service schema panel (bottom left) is augmented to display the interoperability level of each input parameter of a particular service choice. Also, we have developed a ranked list panel that displays the service choices in order of interoperability, and a pop-up compound gauge (Figure 4b) that details the individual input parameters with their subsumption relationships.

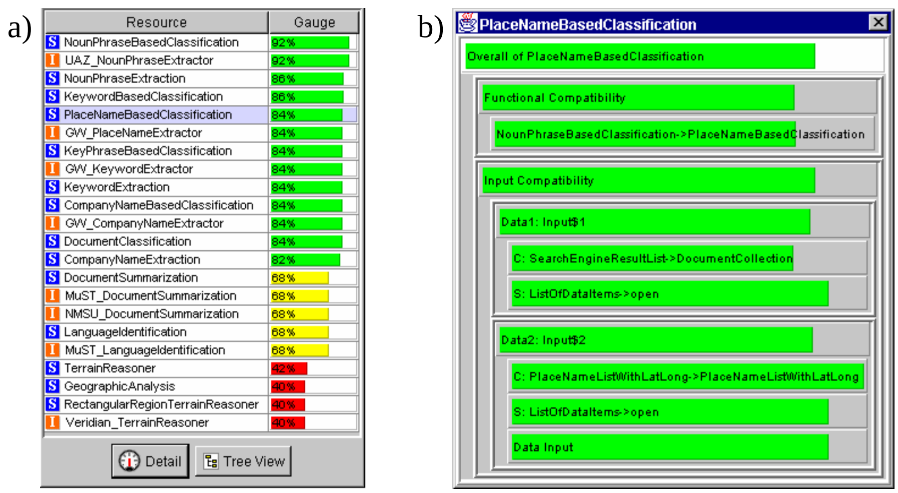


Figure 5 Compatibility Gauges: a) Sorted list of compatible components; b) Compound Gauge

3.4.2 Compatibility gauge

The SIM TBASSCO scripts are design to be portable — scripts generated in one computing service environment can be shared and executed in another environment. However, one problem that arises is that services available in the environment where the script is created may not be available in the execution environment. The Compatibility Gauge helps by finding potential replacement services. Given a particular service in a script, the Compatibility Gauge suggests replacement services based on the semantic closeness of their functionality, input and output.

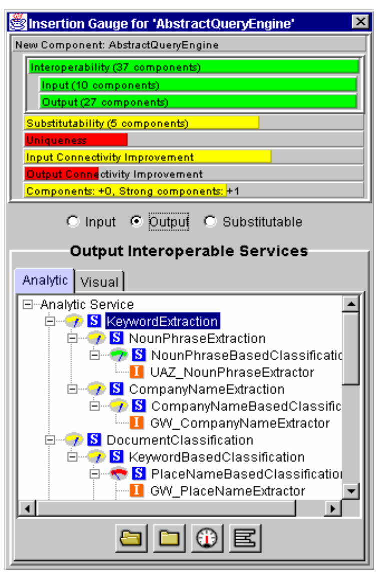
A replacement service, R, is *strictly semantically compatible* with a service, S, if both services have the same functionality AND service R can accept any input that service S can accept, AND service R generates only outputs that service S can generate. For any script, a service may be substituted by a strictly semantically compatible service without semantically affecting that script.

A strictly semantically compatible service is guaranteed to work across all possible scripts, however this compatibility requirement is overly restrictive if the script is known. *Context-dependent semantically compatible* service relaxes this definition by only requiring the replacement service, R, to accept any input that the predecessor services in the script can generate, AND to only generate output that the successor services in the script can accept. In terms of input this means that the replacement service does not have to accept all the input the original service can accept, as long as we can be sure that in the context of this script that such inputs will never be generated. In terms of output this means that the replacement service can generate outputs the original service does not, as long as we can be sure that the successor services in the script can handle the output.

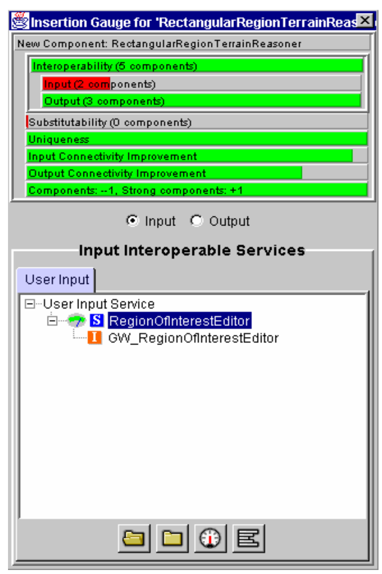
The compatibility gauge is always used in the context of a script, so we adopt the less restrictive context-dependent compatibility. For example Figure 5 shows Compatibility Gauges that are applied to a document classification service, UAZ_NounPhraserExtractor to find out substitutable components of it. Figure 5a is a sorted list of context-dependent semantically compatible services of the classification service. The top ranked compatible services include other phrase-based classification services, such as the place name extractor, keyword extractor and company name extractor. Figure 5b is a compound gauge that shows detail compatibility levels (functional and I/O compatibility) of PlaceNameBasedClassification. However, in the strict semantic compatibility sense the UAZ_NounPhraserExtractor is unique. It is the only service that can be connected to UAZ's SOM clusterer service. If we had used strict semantic compatibility or if there were a SOM clusterer service attached to the UAZ_NounPhraserExtractor, then there would be no alternative compatible service available.

3.4.3 Insertion gauge

GeoWorlds supports a library of component services in a framework intended to help users perform complex information analyses by identifying and applying available component services. Adding additional components to the system is complex, because of the number of existing components and the need for the new components to be interoperable. *Component developers* need help ensuring that offerings conform to appropriate service interfaces, and obey input/output data interchange syntax and semantics.



a)



b)

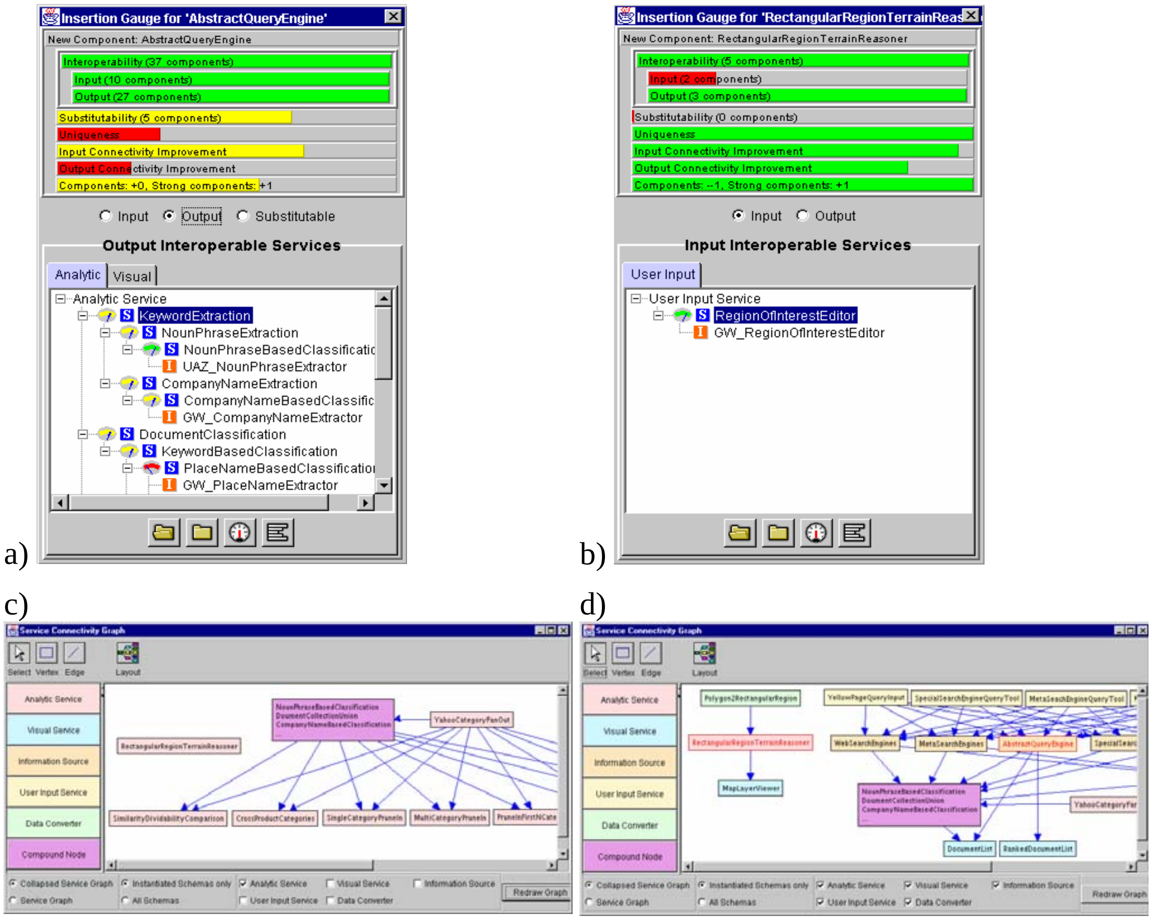


Figure 6 Insertion Gauges: (a) Compound Insertion Gauge Applied to BBN's Abstract Query Engine; (b) Compound Insertion Gauge Applied to Veridian's Terrain Reasoner; (c) Connectivity Graph for Veridian's Terrain Reasoner before adding a converter and visual service; (d) Connectivity Graph for Veridian's Terrain Reasoner after adding the converter and visual service

The Service Insertion Gauge indicates how well new components semantically integrate with existing components in the system, and what other components may be needed to integrate the system more closely. The initial implementation uses the Interoperability Gauge to measure how many existing services understand the output of the new service, and how many existing services are able to generate output that the new services can accept.

GeoWorlds provides us with a testbed for evaluating the utility and effectiveness of these tools. Recently, two new components, developed elsewhere, have been added to the GeoWorlds system. This exercise has provided us with an opportunity to test our Insertion Gauge.

BBN's Abstract Query Engine provides a new service that is similar to query engines already present in GeoWorlds. As such, we would predict that it should score high on interoperability, since provisions for interfacing with other query engines have already been implemented. Application of the Insertion Gauge (Figure 6a) yielded a high score, indicating that insertion would be relatively easy.

Veridian's Terrain Reasoner offers a service that does not resemble any preexisting GeoWorlds components. It computes the difficulty of traversing land on foot. Figure 6c shows the service connectivity graph of the Terrain Reason when it was first introduced into GeoWorlds. It is an isolated node without any other components connected to it. Additional converters, input services and viewers are needed to properly invoke it. Figure 6d shows the service connectivity graph after additional interoperable services are added. However, even with the additional services the Terrain Reasoner still has low degree of service connectivity, and it is still separated from most of the services. The Insertion Gauge gives a lower overall score to the Terrain Reasoner, indicating that the effort to integrate this component into GeoWorlds would be relatively higher, and that interoperation with other components would be less. On the other hand, the fact that the Terrain Reasoner adds a service that did not exist previously is reflected in the creation of its own category within the subsumption hierarchy; this yields a high score for the "uniqueness" dimension (Figure 6b).

4 Results and Discussion

Technical Accomplishments

- Implemented multi-level architectural views that provide visibility into distributed software systems and that identify control points to adjust their behavior thereby enabling dynamic, rapid response to faults.
- Developed dataflow architecture view-based semantic adaptations that reformulate software applications thereby enabling programmers and advanced end users during runtime to substitute alternative software services for non-functioning or unavailable service (92% percent speedup from hours to minutes).
- Developed runtime system architecture view-based transformations that dynamically adjust server deployments thereby enabling system administrators to create additional servers to accommodate increased demand, and to migrate servers from overloaded host to new host (92% percent speed up from hours to minutes).
- Using scripting and semantic gauge technology from Dynamic Assembly for Systems Adaptability's (DASADA) SIM-TBASSCO project we were able to quickly develop the GeoTopics news portal with an estimated 80%-90% speedup over manual development. Using SIM-TBASSCO scripts, the portal collects and analyzes news articles from major English language newspapers. The initial GeoTopics application was created in less than two weeks, and the daily manual labor needed to perform the news analysis and to maintain the GeoTopics website takes less than 30 minutes.

Integration Accomplishments

- Developed joint collaboration and demonstration scenario for IntelliGauge Technology Integration Experiment (TIE) using GeoWorlds/GeoTopics as application test bed
- Developed automatic generation of Acme representation of dataflow architecture view based upon SIM-TBASSCO scripts thereby enabling other DASADA tools to

interface with GeoWorlds.

- Created Acme system architecture view of GeoWorlds and developed system transformation APIs thereby enabling Columbia's KX and CMU's Tailor and Armani tools to automatically balance server loads to improve performance

Defense Impact Accomplishments

- Scripting technology from DASADA's SIM-TBASSCO project was used to perform multiple complex analyses in support of intelligence and operations at Pacific Command (PACOM). These scripts support dynamic assembly of software components in the GeoWorlds information management environment into programs that perform special-purpose information management operations on collections of Web documents

    - The use of SIM-TBASSCO-developed gauges to guide the choice of software components enabled a human expert to rapidly compose GeoWorlds scripts that combined as many as fourteen software components

    - Col. John Cole at PACOM organized five different information collections within two hours using the GeoSpatialTopicAnalysis script; the same tasks performed manually would typically take time measured in weeks

- DASADA's SIM-TBASSCO project successfully installed a version of the GeoWorlds information management environment that has been enhanced with scripting technology, at Joint Forces Command (JFCOM) in Norfolk, VA. Analysts at PACOM and JFCOM are using the DASADA technology to rapidly assemble information analysis programs from GeoWorlds components

    - The GeoWorlds system enhanced by USC ISI SIM-TBASSCO DASADA technology was evaluated by JFCOM Joint Futures Laboratory, as part of its Open Source Information Management (OSIM) Limited Objective Experiments (LOE) initiative, and was recommended for further use at JFCOM

    - The JFCOM OSIM LOE endorsement led to the SIM-TBASSCO-enhanced GeoWorlds system being selected for use and testing in the large-scale Ultimate Vision '01 (UV01) exercise. The system was described to DARPA afterwards as an, "Impressive capability"

    - As a consequence of the UV01 assessment, the SIM-TBASSCO-enhanced GeoWorlds system is currently under consideration by the JFCOM Joint Battle Center for rapid deployment under its charter to deploy near-term interim solutions for pressing long-term JFCOM requirements

- The Joint Experimentation Program (J9) at JFCOM is developing support for very large scale (million-entity) distributed entity-level simulations to enable U.S. military planners to test and improve military doctrines to ensure that U.S. Forces are used effectively. Achieving this 20x scale-up from current capabilities is a huge challenge, which is being addressed by the University of Southern California Information Sciences Institute's (USC ISI) Joint Experimentation on Scalable Parallel Processors (JESPP) project, which executes Joint Semi-Automated Forces (JSAF) simulations on hundreds of interconnected workstations. That effort is profiting from technology produced by USC ISI's SIM-TBASSCO project under DARPA's Dynamic Assembly for Systems Adaptability, Dependability, and Assurance (DASADA) program. SIM-TBASSCO developed "probes and gauges" technology that enables users in real-time to monitor and visualize the run-time behavior of distributed applications within the context of the application's architectural connection topology. These have been ported to JFCOM J9's experimentation environment, where they are used to monitor and analyze multi-processor simulations involving 256 processors distributed over the Internet at 3 sites. The probes and gauges allowed multiple remote users from different locations to instantaneously monitor resource (CPU, memory, network traffic) usage patterns to quickly diagnose faults and inefficiencies.

References.

In-Young Ko, Ke-Thia Yao, and Robert Neches **Dynamic Coordination of Information Management Services for Processing Dynamic Web Content** In Proceedings of The 11th International World Wide Web Conference, May 7-11, 2002, Honolulu, Hawaii, USA.

Ke-Thia Yao, In-Young Ko, Robert Neches, and Robert MacGregor **Semantic Interoperability Scripting and Measurements** In Proceedings of the Working Conference on Complex and Dynamic Systems Architecture, December 2001, Brisbane, Australia.

In-Young Ko, Robert Neches, and Ke-Thia Yao **A Semantic Model and Composition Mechanism for Active Document Collection Templates in Web-based Information Management Systems** Electronic Transactions on Artificial Intelligence (ETAI), Vol. 5, Section D, pp. 55-77, 2001. *(Journal version of the Semantic Web paper)*

In-Young Ko, Robert Neches, and Ke-Thia Yao **Semantically-Based Active Document Collection Templates for Web Information Management Systems** In Proceedings of the International Workshop on the Semantic Web, September 2000, Lisbon, Portugal.

Ke-Thia Yao, In-Young Ko, Ragy Eleish, and Robert Neches **Asynchronous Information Space Analysis Architecture Using Content and Structure Based Service Brokering** In Proceedings of the Fifth ACM International Conference on Digital Libraries (DL 2000), June 2000, San Antonio, Texas.

Ke-Thia Yao, Robert Neches, In-Young Ko, Ragy Eleish, and Sameer Abhinkar **Synchronous and Asynchronous Collaborative Information Space Analysis Tools** In Proceedings of the International Workshop on Collaboration and Mobile Computing (CMC'99), September 1999, University of Aizu, Fukushima, Japan.

Robert Neches, Sameer Abhinkar, Fangqi Hu, Ragy Eleish, In-Young Ko, Ke-Thia Yao, Quan Zhu, and Peter Will **Collaborative Information Space Analysis Tools.** D-Lib Magazine, October 1998. ISSN 1082-9873